

NHẬN XÉT CỦA GIÁO VIÊN

MỤC LỤC

LỜI CẢM ƠN.....	4
I. GIỚI THIỆU.....	5
1. Đơn vị thực tập.....	6
2. Mục đích đề tài.....	6
3. Sơ lược ứng dụng BeliCoffee	7
II. NỘI DUNG ĐỀ TÀI.....	7
1. Giới thiệu hệ điều hành Android.....	7
2. Chi tiết ứng dụng BeliCoffee	9
a) Tài khoản	9
b) Trò chuyện	11
c) Gọi thoại	13
d) Bản đồ và vị trí	23
e) Tin tức và chọn món	24
III. KẾT LUẬN.....	27
1. Hướng phát triển	27
2. Thuận lợi	27
3. Khó khăn	27
4. Kinh nghiệm đạt được.....	27
TÀI LIỆU THAM KHẢO	28

MỤC LỤC HÌNH ẢNH

Hình 1 : Quá trình thực tập tại đơn vị TMA Solutions - Mitel Team	Error!
Bookmark not defined.	
Hình 2 : Giao diện quản lý Auth của admin.....	10
Hình 3: Màn hình đăng nhập, đăng ký và sau khi đăng nhập	10
Hình 4: Giao diện quản lý Storage của admin.....	11
Hình 5: Giao diện quản lý Realtime Database của admin.....	12
Hình 6: Màn hình trò chuyện.....	12
Hình 7: Giao diện quản lý Cloud Messaging của admin.....	13
Hình 8: Thông báo khi nhận được tin nhắn mới	13
Hình 9: Kịch bản cuộc gọi.....	15
Hình 10: Mô hình WebRTC trong thực tế.....	17
Hình 11: Dữ liệu trong SDP	20
Hình 12: Sơ đồ mô tả sự trao đổi.....	21
Hình 13: Màn hình tạo cuộc gọi đi và nhận cuộc gọi đến.....	22
Hình 14: Màn hình cuộc gọi voice call và video call	22
Hình 15: Màn hình bản đồ và vị trí cùng menu chọn nhanh chức năng.....	23
Hình 16: Server của ứng dụng được viết bằng Node.js.....	24
Hình 17: Giao diện quản lý database của MongoAtlas	26
Hình 18: Màn hình tin tức và chọn món.....	26

LỜI CẢM ƠN

Lời đầu tiên, em xin gửi lời cảm ơn chân thành đến thầy cô bộ môn Vật lý Tin học đã tận tình giảng dạy, tổ chức cho em tham gia kì thực tập thực tế này.

Bên cạnh đó, em cũng cảm ơn Mitel Team - TMA Solutions - là đơn vị thực tập, đã tạo điều kiện về cơ sở vật chất, môi trường làm việc, cảm ơn anh hướng dẫn đã tận tình giúp đỡ để em có thể hoàn thành tốt kì thực tập.

Đề tài em được giao là một sự trải nghiệm mới mẻ, giúp em củng cố được những kiến thức cũng như có thêm được những kinh nghiệm bổ ích về môn Lập trình Di động, để có thể áp dụng vào đời sống thực tế cũng như công việc sau này.

Do thời gian gấp rút và kiến thức còn hạn chế, nên chắc chắn sẽ không tránh khỏi sai sót. Vì vậy, em rất mong nhận được những lời góp ý chân thành từ thầy cô và các bạn để có thể hoàn thiện và tiếp tục phát triển đề tài này.

Em xin chân thành cảm ơn!

TP. Hồ Chí Minh, ngày ... tháng ... năm 2019

Sinh viên thực hiện

I. GIỚI THIỆU

1. Đơn vị thực tập

TMA Solutions (gọi tắt TMA, Công ty TNHH Giải Pháp Phần Mềm Tường Minh) là một công ty Việt Nam, có trụ sở tại thành phố Hồ Chí Minh và các văn phòng khác tại Canada, Hoa Kỳ, Ireland, Úc. TMA Solutions kinh doanh các dịch vụ liên quan đến phát triển phần mềm và là một trong những nhà cung cấp phần mềm ra nước ngoài lớn nhất tại Việt Nam.

Mitel Networks Corporation là một công ty viễn thông cung cấp các giải pháp truyền thông hợp nhất cho doanh nghiệp. Công ty trước đây đã sản xuất các hệ thống và ứng dụng TDM PBX, giờ đây tập trung gần như hoàn toàn vào các sản phẩm Thoại qua IP (VoIP). Mitel có trụ sở tại Ottawa, Ontario, Canada, với các văn phòng, đối tác và đại lý trên toàn thế giới.



Hình 1: Quá trình thực tập tại đơn vị Mitel Team - TMA Solutions

2. Mục đích đề tài

Điện thoại thông minh hay smartphone là khái niệm để chỉ loại điện thoại di động thích hợp một nền tảng hệ điều hành di động với nhiều tính năng hỗ trợ tiên tiến về điện toán và có khả năng kết nối với nhiều thiết bị điện tử hiện đại như TV thông minh, máy tính, robot, nhà thông minh, tích hợp hoặc không trí thông minh nhân tạo, dựa trên nền tảng cơ bản của điện thoại di động thông thường (điện thoại phổ thông hay feature phone).

Khái niệm smartphone ra mắt từ những năm 2003-2005. Ban đầu điện thoại thông minh bao gồm các tính năng của điện thoại di động thông thường kết hợp với các thiết bị phổ biến khác như PDA, thiết bị điện tử cầm tay, máy ảnh kỹ thuật số, hệ thống định vị toàn cầu GPS. Điện thoại thông minh hiện đại ngày nay bao gồm hầu như tất cả chức năng của laptop, máy tính như duyệt web, Wi-Fi, đồ họa, văn phòng, chơi game, chụp ảnh, quay phim, video call, định vị toàn cầu, trợ lý ảo, các ứng dụng của bên thứ 3 trên Kho ứng dụng di động và các phụ kiện đi kèm cho máy. Thậm chí một số smartphone cao cấp còn đóng vai trò như một món đồ trang sức đắt tiền, tô điểm cho người chủ nhân.

Trong lịch sử đã từng có nhiều nền tảng hệ điều hành di động cũng như nhiều phong cách thiết kế được sinh ra và bị khai tử. Năm 2007, với sự ra đời của chiếc iPhone thế hệ đầu tiên của Apple với màn hình cảm ứng điện dung, iPhone đã được coi là sự định hình cho kiểu dáng thiết kế điện thoại thông minh hiện đại.

Những điện thoại thông minh phổ biến nhất hiện nay dựa trên nền tảng của 2 hệ điều hành trụ lại thành công duy nhất là Android của Google và iOS của Apple. Trong đó, Android chiếm 87,7% thị phần điện thoại thông minh trên toàn thế giới vào thời điểm quý 2 năm 2017, với tổng cộng 2 tỷ thiết bị đã được kích hoạt và 1,3 triệu lượt kích hoạt mỗi ngày.

Nhận thấy quy mô thị phần của Android, đề tài đã lựa chọn thực hiện trên hệ điều hành này, với mong muốn được tiếp cận với số lượng người sử dụng lớn.

3. Sơ lược ứng dụng BeliCoffee

BeliCoffee là một tổ chức luôn lấy sứ mệnh đào tạo con người làm nền tảng chính, thông qua việc phục vụ những ly cà phê thơm ngon và thuần tự nhiên nhất, đem lại hiệu quả kinh doanh như một phương tiện quan trọng trong việc tạo động lực và cơ hội cho việc hoàn thành sứ mệnh ấy.

Ứng dụng BeliCoffee là một ứng dụng đặt cà phê được viết trên hệ điều hành Android bằng ngôn ngữ Java. Ứng dụng có tích hợp thêm các chức năng như gọi điện, nhắn tin, bản đồ, tin tức,...

II. NỘI DUNG ĐỀ TÀI:

1. Giới thiệu hệ điều hành Android

Android là một hệ điều hành dựa trên nền tảng Linux được thiết kế dành cho các thiết bị di động có màn hình cảm ứng như điện thoại thông minh và máy tính bảng. Ban đầu, Android được phát triển bởi Android, Inc. với sự hỗ trợ tài chính từ Google và sau này được chính Google mua lại vào năm 2005.

Android ra mắt vào năm 2007 cùng với tuyên bố thành lập Liên minh thiết bị cầm tay mở: một hiệp hội gồm các công ty phần cứng, phần mềm, và viễn thông với mục tiêu đẩy mạnh các tiêu chuẩn mở cho các thiết bị di động. Chiếc điện thoại đầu tiên chạy Android được bán vào năm 2008.

Android có mã nguồn mở và Google phát hành mã nguồn theo Giấy phép Apache. Chính mã nguồn mở cùng với một giấy phép không có nhiều ràng buộc đã cho phép các nhà phát triển thiết bị, mạng di động và các lập trình viên nhiệt huyết được điều chỉnh và phân phối Android một cách tự do. Ngoài ra, Android còn có một cộng đồng lập trình viên đông đảo chuyên viết các ứng dụng để mở rộng chức năng của thiết bị, bằng một loại ngôn ngữ lập trình Java có sửa đổi. Tháng 10 năm 2012, có khoảng 700.000 ứng dụng trên Android, và số lượt tải ứng dụng từ Google Play, cửa hàng ứng dụng chính của Android, ước tính khoảng 25 tỷ lượt.

Giao diện người dùng của Android dựa trên nguyên tắc tác động trực tiếp, sử dụng cảm ứng chạm tương tự như những động tác ngoài đời thực như vuốt,

chạm, kéo giãn và thu lại để xử lý các đối tượng trên màn hình. Sự phản ứng với tác động của người dùng diễn ra gần như ngay lập tức, nhằm tạo ra giao diện cảm ứng mượt mà, thường dùng tính năng rung của thiết bị để tạo phản hồi rung cho người dùng. Những thiết bị phần cứng bên trong như gia tốc kế, con quay hồi chuyển và cảm biến khoảng cách được một số ứng dụng sử dụng để phản hồi một số hành động khác của người dùng, ví dụ như điều chỉnh màn hình từ chế độ hiển thị dọc sang chế độ hiển thị ngang tùy theo vị trí của thiết bị, hoặc cho phép người dùng lái xe đua bằng xoay thiết bị, giống như đang điều khiển vô-lăng.

Các thiết bị Android sau khi khởi động sẽ hiển thị màn hình chính, điểm khởi đầu với các thông tin chính trên thiết bị, tương tự như khái niệm desktop (bàn làm việc) trên máy tính để bàn. Màn hình chính Android thường gồm nhiều biểu tượng (icon) và tiện ích (widget); biểu tượng ứng dụng sẽ mở ứng dụng tương ứng, còn tiện ích hiển thị những nội dung sống động, cập nhật tự động như dự báo thời tiết, hộp thư của người dùng, hoặc những mẫu tin thời sự ngay trên màn hình chính. Màn hình chính có thể gồm nhiều trang xem được bằng cách vuốt ra trước hoặc sau, mặc dù giao diện màn hình chính của Android có thể tùy chỉnh ở mức cao, cho phép người dùng tự do sắp đặt hình dáng cũng như hành vi của thiết bị theo sở thích. Những ứng dụng do các hãng thứ ba có trên Google Play và các kho ứng dụng khác còn cho phép người dùng thay đổi "chủ đề" của màn hình chính, thậm chí bắt chước hình dáng của hệ điều hành khác như Windows Phone chẳng hạn. Phần lớn những nhà sản xuất, và một số nhà mạng, thực hiện thay đổi hình dáng và hành vi của các thiết bị Android của họ để phân biệt với các hãng cạnh tranh.

Ở phía trên cùng màn hình là thanh trạng thái, hiển thị thông tin về thiết bị và tình trạng kết nối. Thanh trạng thái này có thể "kéo" xuống để xem màn hình thông báo gồm thông tin quan trọng hoặc cập nhật của các ứng dụng, như email hay tin nhắn SMS mới nhận, mà không làm gián đoạn hoặc khiến người dùng cảm thấy bất tiện. Trong các phiên bản đời đầu, người dùng có thể nhấn vào thông báo để mở ra ứng dụng tương ứng, về sau này các thông tin cập nhật được bổ sung thêm tính năng, như có khả năng lập tức gọi ngược lại khi có cuộc gọi nhỡ

mà không cần phải mở ứng dụng gọi điện ra. Thông báo sẽ luôn nằm đó cho đến khi người dùng đã đọc hoặc xóa nó đi.

2. Chi tiết ứng dụng BeliCoffee

Ứng dụng BeliCoffee có các tính năng chính:

- Tài khoản: Đăng ký, Đăng nhập, Đăng xuất, Quên mật khẩu.
- Trò chuyện.
- Gọi thoại.
- Bản đồ và vị trí.
- Tin tức, chọn món.

a) Tài khoản

Ứng dụng sử dụng Firebase như một server, một back end. Firebase là một nền tảng phát triển ứng dụng di động và web được phát triển bởi Firebase, Inc. vào năm 2011, sau đó được Google mua lại vào năm 2014. Tính đến tháng 10 năm 2018, nền tảng Firebase có 18 sản phẩm, được sử dụng bởi 1,5 triệu ứng dụng. Trong phạm vi tài, ứng dụng chỉ sử dụng 4 sản phẩm của Firebase, bao gồm: FirebaseAuth, Firebase Storage, Firebase Realtime Database và Firebase Cloud Messaging.

Firebase Auth là một dịch vụ có thể xác thực người dùng chỉ sử dụng mã phía máy khách. Nó hỗ trợ các nhà cung cấp đăng nhập xã hội Facebook, GitHub, Twitter và Google (và Google Play Games). Ngoài ra, nó bao gồm một hệ thống quản lý người dùng, theo đó các nhà phát triển có thể cho phép xác thực người dùng bằng email và mật khẩu đăng nhập được lưu trữ với Firebase.

Ứng dụng sử dụng Firebase Auth để quản lý, cho phép người dùng đăng ký tài khoản, đăng nhập, đăng xuất trong ứng dụng, cũng như lấy lại mật khẩu qua email mà người dùng đã đăng ký trước đó.

The screenshot shows the Firebase Authentication console under the 'Users' tab. It displays a table of user information with columns: Identifier, Providers, Created, Signed In, and User UID. The table lists several users, each associated with a Gmail account and a unique User UID.

Identifier	Providers	Created	Signed In	User UID
tung@gmail.com	✉	Jul 26, 2019	Sep 6, 2019	4cc5yhkAXGapiVTE3gXimY59b7N2...
daithinh@gmail.com	✉	Aug 20, 2019	Aug 20, 2019	6u95MTNIM3eeOzo540qf7Zlw2t2...
tonbach18598@gmail.com	✉	Sep 4, 2019	Sep 6, 2019	7kYncMG03h0Gump346De9H93Jq...
ha@gmail.com	✉	Jul 26, 2019	Jul 26, 2019	EnESWBjSDQZ6fgKm0uV0zujMoc...
bach2@gmail.com	✉	Sep 4, 2019	Sep 4, 2019	InhncwurnwYMcdVBuRhgTLbfJhN2...
thinh@gmail.com	✉	Jul 26, 2019	Sep 25, 2019	JV5hvs82uEQ16mYvbd821yNmVZ...
thinh1@gmail.com	✉	Aug 20, 2019	Aug 20, 2019	TeWiXea3Ihr0eidE7cuqSCYAGz2...

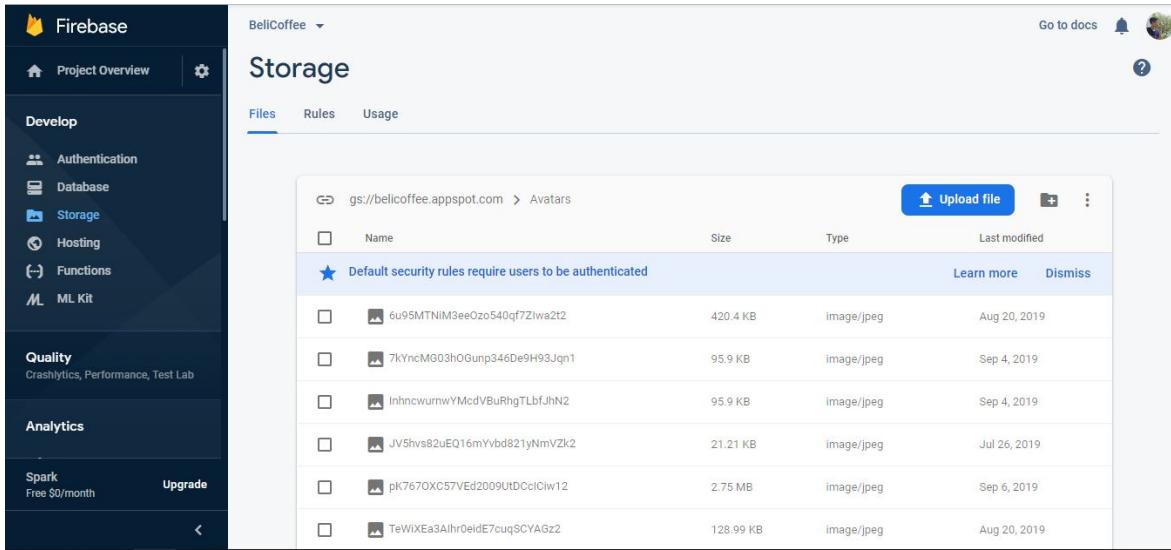
Hình 2: Giao diện quản lý Auth của admin

The screenshots illustrate the Beli Coffee mobile application's user interface. The first screen shows the login page with fields for Email and Password, and links for forgot password and sign up. The second screen shows the registration page with fields for Email, Password, Username, and Phone Number. The third screen shows the user's profile dashboard with options like Home, Order history, Favorites, Chat, and Logout.

Hình 3: Màn hình đăng nhập, đăng ký và sau khi đăng nhập

Firebase Storage cung cấp tải lên và tải xuống tệp an toàn cho các ứng dụng Firebase, bất kể chất lượng mạng. Nhà phát triển có thể sử dụng nó để lưu trữ hình ảnh, âm thanh, video hoặc nội dung khác do người dùng tạo. Lưu trữ Firebase được hỗ trợ bởi Google Cloud Storage.

Ứng dụng sử dụng Firebase Storage để lưu trữ ảnh đại diện của người dùng và hình ảnh của món ăn, tin tức.



Hình 4: Giao diện quản lý Storage của admin

b) Trò chuyện

Firebase Realtime Database cung cấp một cơ sở dữ liệu thời gian thực và phụ trợ như một dịch vụ. Dịch vụ này cung cấp cho các nhà phát triển ứng dụng một API cho phép dữ liệu ứng dụng được đồng bộ hóa giữa các máy khách và được lưu trữ trên đám mây của Firebase. Công ty cung cấp các thư viện máy khách cho phép tích hợp với các ứng dụng Android, iOS, JavaScript, Java, Objective-C, Swift và Node.js. Cơ sở dữ liệu cũng có thể truy cập thông qua API REST và các ràng buộc cho một số khung JavaScript như AngularJS, React, Ember.js và Backbone.js. API REST sử dụng giao thức Sự kiện gửi máy chủ, là API để tạo kết nối HTTP để nhận thông báo đẩy từ máy chủ. Các nhà phát triển sử dụng cơ sở dữ liệu thời gian thực có thể bảo mật dữ liệu của họ bằng cách sử dụng các quy tắc bảo mật được thi hành bởi phía máy chủ của công ty.

Ứng dụng sử dụng Firebase Realtime Database để lưu trữ thông tin cá nhân người dùng, nội dung các cuộc trò chuyện cũng như lịch sử vị trí của người dùng. Cơ sở dữ liệu được lưu trữ dưới dạng cây JSON, có thể truy xuất một cách nhanh chóng.

The screenshot shows the Firebase Realtime Database interface. On the left, there's a sidebar with project settings like Project Overview, Develop (Authentication, Database, Storage, Hosting, Functions, ML Kit), Quality (Crashlytics, Performance, Test Lab), and Extensions. The main area is titled 'Database' under 'Realtime Database'. It has tabs for Data, Rules, Backups, and Usage. Below is a tree view of the database structure under 'belicoffee/Chats'. A specific message node is expanded, showing a timestamp of '15:08', a receiver ID, a sender ID, and the message content 'hi'. There are also other message nodes below it.

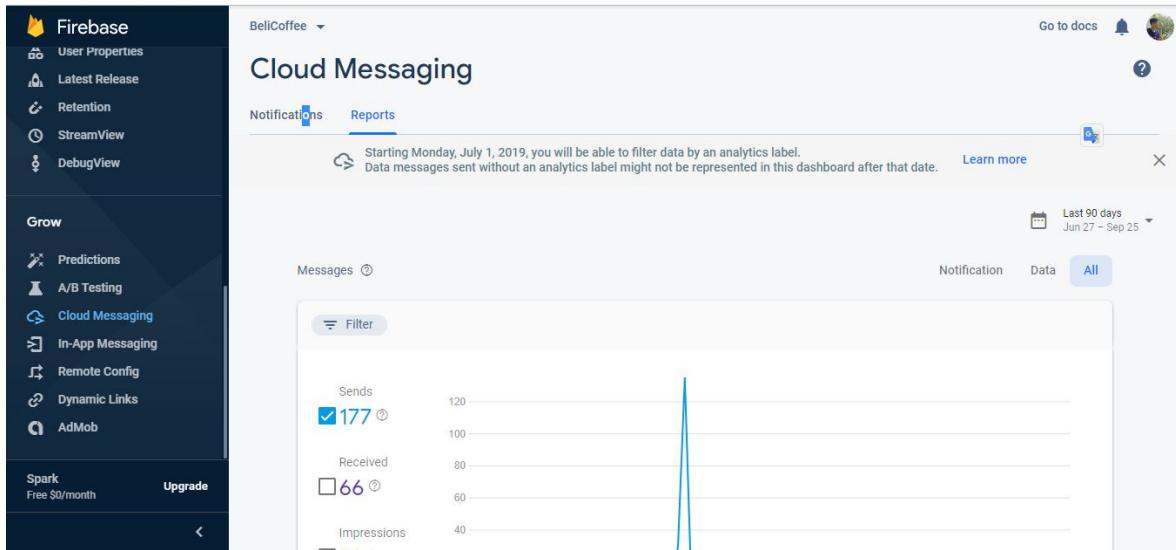
Hình 5: Giao diện quản lý Realtime Database của admin



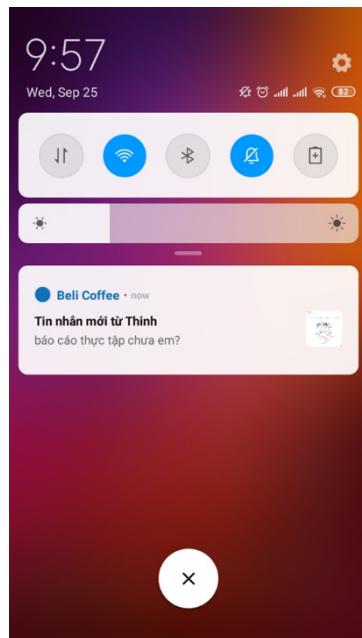
Hình 6: Màn hình trò chuyện

Firebase Cloud Messaging (FCM), trước đây gọi là Google Cloud Messaging (GCM), là một giải pháp đám mây đa nền tảng cho các tin nhắn và thông báo cho Android, iOS và các ứng dụng web, hiện có thể được sử dụng miễn phí.

Ứng dụng sử dụng Firebase Cloud Messaging để thông báo đến người dùng mỗi khi có thông báo từ phía server (tin tức, khuyến mãi,...), cũng như thông báo khi người dùng nhận được tin nhắn mới từ người dùng khác.



Hình 7: Giao diện quản lý Cloud Messaging của admin



Hình 8: Thông báo khi nhận được tin nhắn mới

c) Gọi thoại

Ứng dụng sử dụng WebRTC để thực hiện chức năng gọi thoại (voice call và video call)

WebRTC (Web Real-Time Communication) là một dự án nguồn mở, miễn phí, cung cấp các trình duyệt web và ứng dụng di động với giao tiếp thời gian thực (RTC) thông qua các giao diện lập trình ứng dụng đơn giản (API). Nó cho phép giao tiếp âm thanh và video hoạt động bên trong các trang web bằng cách cho phép giao tiếp ngang hàng trực tiếp, loại bỏ nhu cầu cài đặt plugin hoặc tải xuống ứng dụng gốc. Được hỗ trợ bởi Apple, Google, Microsoft, Mozilla và Opera, WebRTC đang được chuẩn hóa thông qua World Wide Web Consortium (W3C) và Lực lượng đặc nhiệm kỹ thuật Internet (IETF).

Nhiệm vụ của nó là "cho phép các ứng dụng RTC phong phú, chất lượng cao được phát triển cho trình duyệt, nền tảng di động và thiết bị IoT và cho phép tất cả chúng giao tiếp thông qua một bộ giao thức chung".

i. Giao tiếp Peer-To-Peer (P2P)

Để có thể giao tiếp lẫn nhau thông qua trình duyệt web, mỗi trình duyệt của user phải thực hiện những bước sau đây:

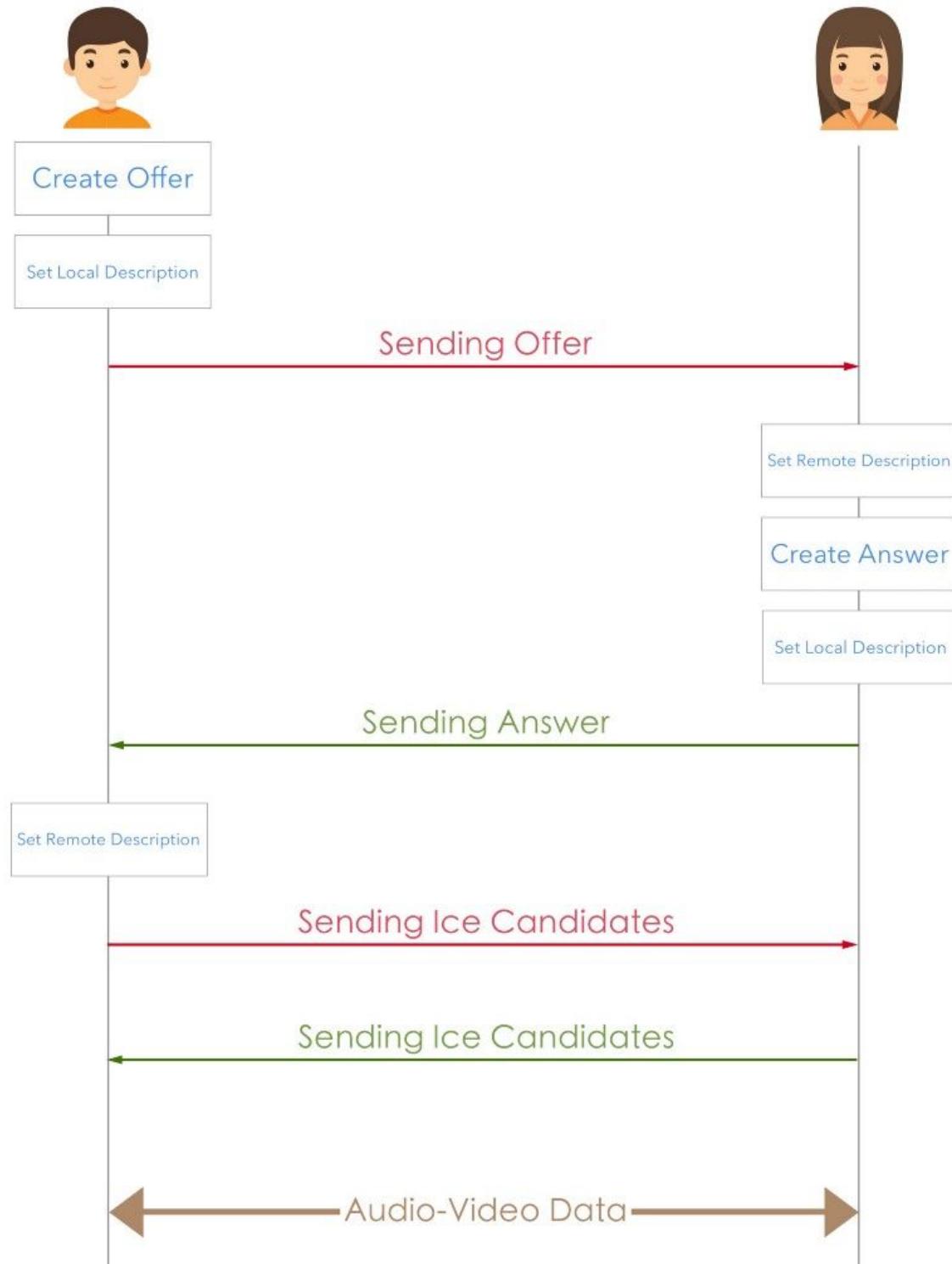
- Đồng ý để bắt đầu giao tiếp
- Biết cách xác định vị trí của đối tượng
- Vượt qua an ninh và tường lửa bảo vệ
- Chuyển giao tất cả các giao tiếp đa phương tiện theo real-time

Một trong số những thách thức lớn nhất liên quan đến các giao tiếp P2P dựa trên trình duyệt là làm sao để biết vị trí & thiết lập 1 kết nối socket mạng (network socket connection) với 1 trình duyệt khác để vận chuyển dữ liệu 2 chiều. Ta sẽ xem xét những khó khăn liên quan đến việc thiết lập kết nối này.

Khi ứng dụng của bạn cần dữ liệu hoặc tài nguyên, nó sẽ lấy về từ các server. Tuy nhiên, nếu bạn muốn tạo ra 1 ứng dụng video chat chẳng hạn, bằng cách kết nối trực tiếp đến trình duyệt của người khác - thì đây là vấn đề, vì bạn không biết địa chỉ bởi vì trình duyệt của người kia không phải là 1 web server. Vì vậy để có thể thiết lập 1 kết nối P2P ta cần rất nhiều thứ.

ii. Kịch bản cuộc gọi:

Một chàng trai muốn gọi cho bạn gái của mình qua video call, và chàng trai đã chọn chức năng gọi thoại. Sau đây là kịch bản:



Hình 9: Kịch bản cuộc gọi

- Đầu tiên, ứng dụng BeliCoffee sẽ tạo một kết nối ngang hàng và một offer SDP. Offer này chứa data về cuộc gọi và được sử dụng để nhận dạng các codec và các đối tượng khác của kết nối ngang hàng.
- Offer này sau đó sẽ được lưu lại như là một 'dữ liệu về local' tại thời điểm kết nối và sau đó gửi đến người nhận qua một cơ chế thông báo (có thể sử dụng socket)
- Khi bên cô gái nhận được tín hiệu, ứng dụng bên đó sẽ biết được có một cuộc gọi được thiết lập, nó sẽ lưu offer như là một 'dữ liệu remote' và tạo một Answer SDP.
- Answer SDP này tương tự như Offer SDP của người gọi, chứa các dữ liệu chi tiết cho người dùng ngang hàng.
- Ứng dụng tại phía cô gái lưu Answer SDP như là 'dữ liệu về local' của nó và gửi nó thông qua các tín hiệu thông báo (socket) cho chàng trai.
- Chàng trai nhận được câu trả lời và lưu nó là 'dữ liệu remote'.
- Chàng trai và cô gái sau đó truyền các Ice Candidate liên quan đến họ thông qua các kênh báo hiệu. Khi nhận được các Ice Candidate này, kết nối sẽ thêm vào PeerConnection.
- Sau khi việc truyền các Candidate hoàn thành, hai bên sẽ bắt đầu truyền media data với nhau. Việc truyền data này thông qua RTP và WebRTC framework.

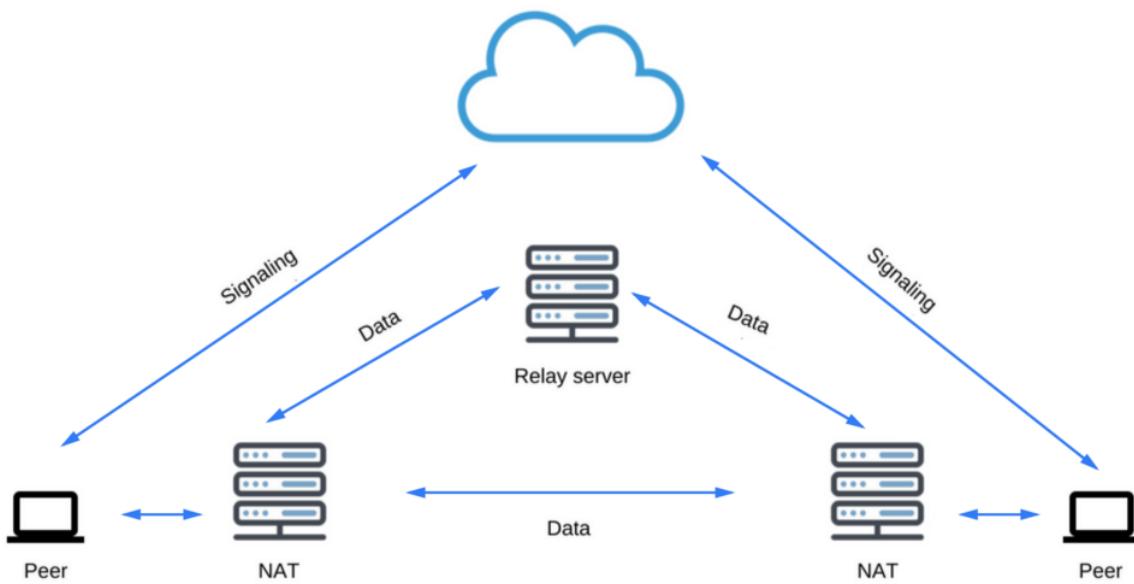
iii. WebRTC trong thế giới thực

Trong thế giới thực thì WebRTC cần có server, tuy nhiên thực tế lại đơn giản hơn:

- Các user tự khám phá ra đối tác của họ và trao đổi các chi tiết chặng hạn như tên.
- Các ứng dụng WebRTC phía client (các peer) trao đổi thông tin mạng.
- Các peer trao đổi dữ liệu về media chặng hạn như định dạng hình ảnh và độ phân giải.
- Các ứng dụng WebRTC phía client di chuyển xuyên qua các cổng NAT và tường lửa.

Nói cách khác, WebRTC cần phải có 4 tính năng ở phía server:

- User khám phá ra và giao tiếp.
- Signaling
- Di chuyển NAT/tường lửa
- Các server chuyển tiếp trong trường hợp giao tiếp peer-to-peer thất bại.



Hình 10: Mô hình WebRTC trong thực tế

iv. Tường lửa và NAT Traversal

Thông thường thì máy tính của bạn không có địa chỉ IP public tĩnh. Lý do là máy tính của bạn phải nấp đằng sau tường lửa và thiết bị NAT (Network access translation - Bộ phiên dịch truy cập mạng).

Thiết bị NAT sẽ dịch địa chỉ IP cá nhân từ bên trong tường lửa thành địa chỉ IP công khai (public-facing IP). Ta cần các thiết bị NAT để bảo mật và giải quyết sự giới hạn của IPv4 đối với những địa chỉ IP công khai sẵn có. Đó là lý do tại sao webapp không nên giả định rằng thiết bị hiện tại có 1 địa chỉ IP public tĩnh.

Nếu như bạn đang ở trong môi trường công cộng và kết nối vào mạng WiFi, máy tính của bạn sẽ được gán 1 địa chỉ IP mà nó chỉ tồn tại đằng sau NAT. Giả sử IP là 172.0.23.4, tuy nhiên, với thế giới bên ngoài, địa chỉ IP của bạn có thể mang giá trị khác, ví dụ 164.53.27.98. Vì vậy, thế giới bên ngoài sẽ thấy các

request của bạn đến từ địa chỉ 164.53.27.98 nhưng thiết bị NAT sẽ đảm bảo các response cho những request (được gửi từ máy của bạn) sẽ trả về đúng chỗ là 172.0.23.4. Lưu ý rằng ngoài địa chỉ IP thì cổng (port) cũng là điều kiện cần thiết cho các giao tiếp mạng.

Do có sự tham gia của các thiết bị NAT, trình duyệt của bạn cần tìm được địa chỉ IP của máy tính có trình duyệt mà bạn muốn giao tiếp.

Đến đây thì ta phải nhờ đến các server STUN (Session Traversal Utilities for NAT - Tiện ích truyền tải theo phiên cho NAT) và TURN (Traversal Using Relays around NAT - Truyền tải sử dụng điểm chuyển tiếp vòng quanh NAT).

Để các công nghệ WebRTC hoạt động được, đầu tiên thì 1 request hỏi địa chỉ IP public của bạn sẽ được gửi đến server STUN. Bạn cứ nghĩ theo hướng kiểu như máy tính đang tạo truy vấn đến 1 server từ xa để hỏi về địa chỉ IP mà server đó nhận câu truy vấn là bao nhiêu. Server từ xa sẽ trả về địa chỉ IP mà nó thấy. Nói ngắn gọn là máy tính của bạn "hỏi" 1 server từ xa địa chỉ IP của chính máy bạn là bao nhiêu.

ICE (Interactive Connectivity Establishment - Thiết lập kết nối tương tác) là 1 giao thức để kết nối các peer, chẳng hạn như 2 ứng dụng video chat client. Khi khởi tạo, ICE sẽ thử kết nối trực tiếp đến các peer với độ trễ thấp nhất có thể thông qua UDP. Trong tiến trình này, các server STUN có 1 tác vụ duy nhất: kích hoạt 1 peer phía sau NAT để tìm địa chỉ public và số port.

Nếu như UDP thất bại, ICE sẽ thử TCP: đầu tiên là HTTP, sau đó là HTTPS. Nếu kết nối trực tiếp thất bại - cụ thể là bởi vì dịch chuyển NAT & tường lửa mức độ doanh nghiệp - ICE sẽ dùng 1 server TURN trung gian (điểm chuyển tiếp). Nói cách khác, ICE đầu tiên sẽ dùng STUN với UDP để kết nối trực tiếp các peer với nhau, nếu thất bại, nó sẽ đổi kế hoạch sang dùng server chuyển tiếp TURN. Cụm từ "tìm kiếm ứng viên" nhắc đến quá trình tìm kiếm các giao diện mạng và port.

Nếu như tiến trình đồng ý chấp nhận ứng viên ICE tốt nhất bị thất bại, thỉnh thoảng nguyên nhân là do tường lửa hoặc kỹ thuật NAT đang dùng, thì giải pháp

dự phòng sau đó là sử dụng 1 server TURN dưới dạng 1 điểm chuyển tiếp thay thế. Tiến trình này về cơ bản sẽ dùng 1 server hoạt động như người trung gian và nó chuyển tiếp bất kỳ dữ liệu nào truyền qua lại giữa các peer. Lưu ý rằng đây không phải là giao tiếp peer-to-peer thực thụ trong đó các peer truyền dữ liệu 2 chiều trực tiếp đến với nhau.

Khi sử dụng giải pháp dự phòng TURN để giao tiếp, mỗi peer sẽ không cần phải biết làm thế nào để liên lạc và truyền dữ liệu đến bên kia. Thay vào đó, nó cần biết server TURN public nào để gửi và nhận dữ liệu đa phương tiện theo thời gian thực xuyên suốt phiên giao tiếp.

Điều quan trọng cần phải hiểu rằng đây chắc chắn là 1 dạng "kế hoạch dự phòng" và chỉ dùng khi không còn cách nào khác. Các server TURN phải khá vững chắc, có băng thông rộng, khả năng xử lý và có thể xử lý 1 lượng lớn dữ liệu tiềm tàng. Cách sử dụng server TURN vì thế rõ ràng là sẽ phát sinh thêm chi phí và sự phức tạp.

Giả sử tiến trình này hoạt động bình thường và bạn nhận được địa chỉ IP public của mình cũng như số port, bạn sẽ có thể nói với những peer ngang hàng khác làm thế nào để kết nối trực tiếp đến bạn. Những peer này cũng có thể làm cùng 1 việc là sử dụng STUN & server TURN và nói cho bạn biết nên liên lạc đến địa chỉ nào.

v. *Tín hiệu, phiên, giao thức (Signaling, Sessions, Protocols)*

Quá trình khám phá thông tin mạng được mô tả ở trên chỉ là 1 phần của chủ đề về Signaling to bự hơn nhiều, trong trường hợp của WebRTC thì phần signaling này dựa trên 1 chuẩn JSEP (**Javascript Session Establishment Protocol** - Giao thức thiết lập phiên của Javascript). Signaling bao gồm cả khám phá mạng (network discovery) và NAT Traversal, tạo và quản lý phiên, bảo mật giao tiếp, siêu dữ liệu (metadata) và phối hợp khả năng của media, xử lý lỗi.

Để kết nối có thể hoạt động, peer thu được những điều kiện về local media cho metadata (ví dụ: những khả năng về kích thước và kiểu codec) và gom góp các địa chỉ mạng có thể có cho host của ứng dụng. Cơ chế signaling dùng để

truyền tới/lui những thông tin quan trọng này không được định nghĩa trong API của WebRTC.

Signaling không được quy định bởi chuẩn WebRTC và nó không được triển khai bằng API của nó để cho phép sử dụng một cách linh động các công nghệ và giao thức cần thiết. Các nhà phát triển WebRTC sẽ đối phó với signaling và server xử lý signaling.

Giả sử app WebRTC dựa trên trình duyệt của bạn có thể xác định được địa chỉ IP public của nó bằng cách sử dụng STUN như đã nói ở trên, bước tiếp theo thực sự là 1 màn đàm phán & thiết lập phiên kết nối đến với peer.

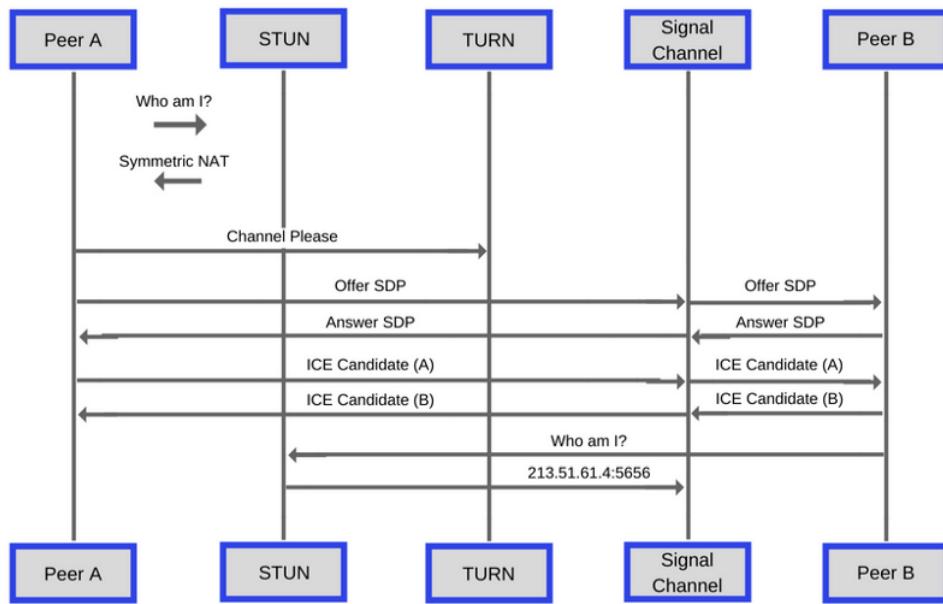
Phần đàm phán và thiết lập khởi tạo phiên xảy ra khi dùng 1 giao thức signaling/giao tiếp được đặc tả trong các giao tiếp đa phương tiện. Giao thức này cũng chịu trách nhiệm cho việc điều hành các quy định trong đó phiên được quản lý và hủy bỏ.

Một trong số các giao thức như vậy có tên là **Session Initiation Protocol** (SIP - Giao thức khởi tạo phiên). Lưu ý rằng do sự linh động của WebRTC signaling, SIP không phải là giao thức signaling duy nhất có thể dùng. Giao thức signaling được chọn phải hoạt động với 1 giao thức ở tầng ứng dụng gọi là **Session Description Protocol** (SDP - Giao thức mô tả phiên), giao thức này được sử dụng trong trường hợp của WebRTC. Tất cả các metadata đa phương tiện cụ thể được truyền đi bằng giao thức SDP này.

```
v=0
o=- 3883943731 1 IN IP4 127.0.0.1
s=
t=0 0
a=group:BUNDLE audio video
m=audio 1 RTP/SAVPF 103 104 0 8 106 105 13 126
// ...
a:ssrc:2223794119 label:H4fjnMzxy3dPIgQ7HxuCTLb4wLLLeRHnFxh810
```

Hình 11: Dữ liệu trong SDP

Bất kỳ peer nào (ví dụ: ứng dụng tận dụng WebRTC) thử giao tiếp với 1 peer khác đều sinh ra một tập các ứng viên giao thức ICE. Những ứng viên này biểu diễn 1 bộ kết hợp của địa chỉ IP, port, giao thức giao vận được dùng. Lưu ý rằng một máy tính có thể có nhiều giao diện mạng (không dây, có dây, vân vân), vì thế có thể được gán nhiều địa chỉ IP cho mỗi giao diện.



Hình 12: Sơ đồ mô tả sự trao đổi

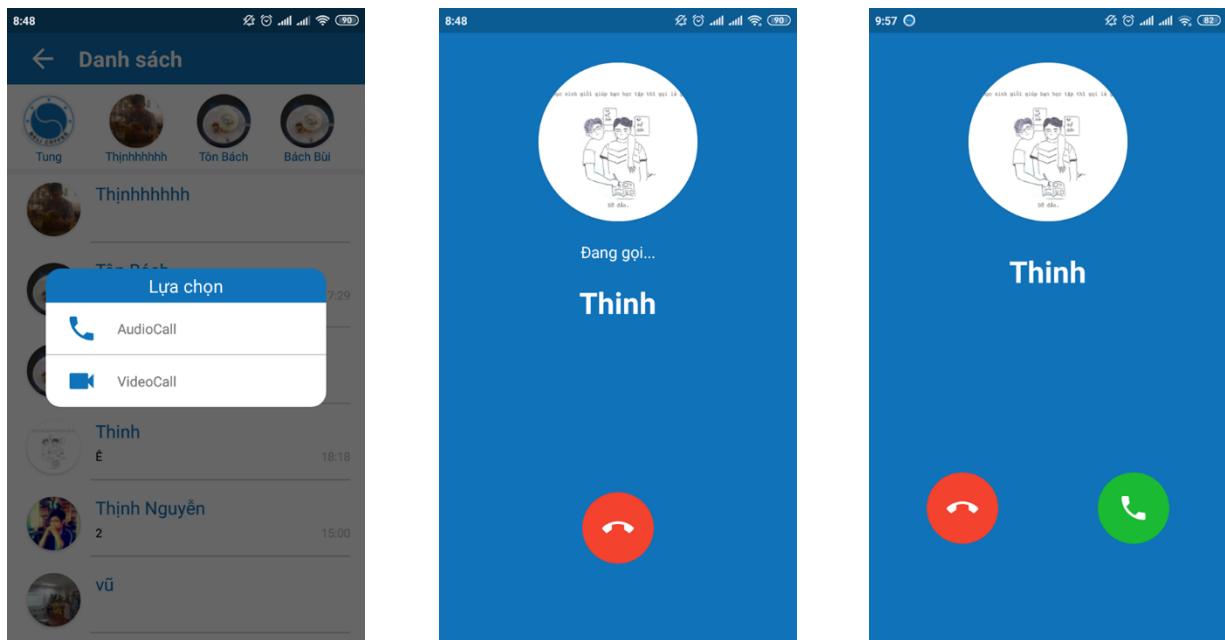
vi. Các API của WebRTC

Có 3 mục phân loại API chính trong WebRTC:

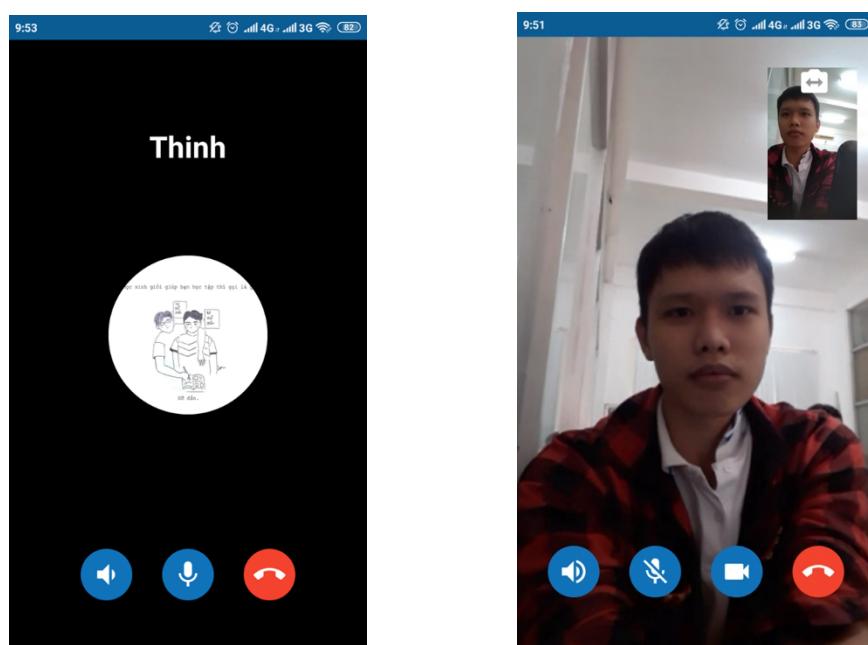
- Media Capture and Streams (luồng media và chụp media) - cho phép bạn truy xuất vào các thiết bị đầu vào, chẳng hạn như microphone hay web camera. API cho phép bạn lấy 1 luồng media từ các thiết bị đó.
- RTCPeerConnection - dùng những API này, bạn có thể gửi theo thời gian thực 1 luồng âm thanh & hình ảnh đã bắt được thông qua internet đến 1

endpoint WebRTC khác. Bạn có thể tạo ra kết nối giữa máy local và peer từ xa. Nó cũng cung cấp các phương thức để kết nối đến 1 peer từ xa, duy trì và kiểm soát kết nối & đóng kết nối 1 khi ta không cần đến nó nữa.

- RTCDataChannel - API này cho phép bạn truyền dữ liệu tùy ý. Mỗi kênh dữ liệu được liên kết với 1 RTCPeerConnection.



Hình 13: Màn hình tạo cuộc gọi đi và nhận cuộc gọi đến



Hình 14: Màn hình cuộc gọi voice call và video call

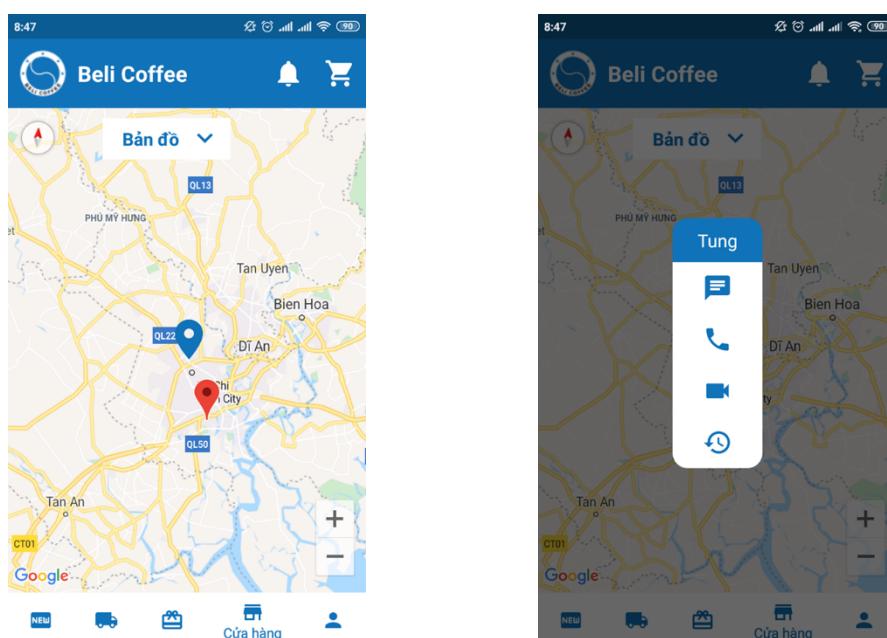
d) Bản đồ và vị trí

Google Maps là một dịch vụ bản đồ số được Google phát triển với mục đích thay thế cho các loại bản đồ giấy thông thường. Giờ đây chỉ bằng chiếc smartphone nhỏ gọn bạn có thể tự do lựa chọn những địa điểm mà mình muốn đến và nhanh chóng tiếp cận được những dịch vụ xung quanh các địa điểm đó.

Để sử dụng Google Maps một cách chính xác nhất bạn cần GPS - hệ thống định vị toàn cầu giúp bạn có thể biết rõ vị trí hiện tại của bản thân và thông qua GPS con người có thể dễ dàng xác định được phương hướng và đường đi một cách nhanh chóng nhất có thể.

Ứng dụng tích hợp Google Maps nhằm hiển thị vị trí hiện tại của bản thân, cũng như của bạn bè trong danh sách. Bên cạnh đó, ứng dụng cũng có chức năng hiển thị lịch sử vị trí của bản thân hoặc bạn bè với các lựa chọn 3 ngày, 7 ngày và 30 ngày.

Ngoài ra, ứng dụng có hỗ trợ thao tác gọi thoại và nhắn tin khi nhấn vào vị trí của bạn bè trên bản đồ, điều này giúp cho việc liên lạc với bạn bè trở nên dễ dàng và nhanh chóng.



Hình 15: Màn hình bản đồ và vị trí cùng menu chọn nhanh chúc năng

e) Tin tức và chọn món

Ứng dụng sử dụng server được viết bằng Node.js với cơ sở dữ liệu MongoDB (là một NoSQL) để thực hiện việc hiển thị tin tức và danh sách các món. Các phương thức hỗ trợ cho admin bao gồm GET, POST, PUT và DELETE.

i. Node.js

- Nodejs là một nền tảng (platform) phát triển độc lập được xây dựng ở trên Javascript Runtime của Chrome mà chúng ta có thể xây dựng được các ứng dụng mạng một cách nhanh chóng và dễ dàng mở rộng.
- Nodejs được xây dựng và phát triển từ năm 2009, bảo trợ bởi công ty Joyent, trụ sở tại California, Hoa Kỳ.
- Phần Core bên dưới của Nodejs được viết hầu hết bằng C++ nên cho tốc độ xử lý và hiệu năng khá cao.
- Nodejs tạo ra được các ứng dụng có tốc độ xử lý nhanh, realtime thời gian thực.
- Nodejs áp dụng cho các sản phẩm có lượng truy cập lớn, cần mở rộng nhanh, cần đổi mới công nghệ, hoặc tạo ra các dự án Startup nhanh nhất có thể.



Hình 16: Server của ứng dụng được viết bằng Node.js

ii. NoSQL

- NoSQL là 1 dạng CSDL mã nguồn mở và được viết tắt bởi: None-Relational SQL hay có nơi thường gọi là Not-Only SQL.

- NoSQL được phát triển trên Javascript Framework với kiểu dữ liệu là JSON và dạng dữ liệu theo kiểu key và value.
- NoSQL ra đời như là 1 mảnh vá cho những khuyết điểm và thiếu sót cũng như hạn chế của mô hình dữ liệu quan hệ RDBMS (Relational Database Management System - Hệ quản trị cơ sở dữ liệu quan hệ) về tốc độ, tính năng, khả năng mở rộng,....
- Với NoSQL bạn có thể mở rộng dữ liệu mà không lo tới những việc như tạo khóa ngoại, khóa chính, kiểm tra ràng buộc,....
- NoSQL bỏ qua tính toàn vẹn của dữ liệu và transaction để đổi lấy hiệu suất nhanh và khả năng mở rộng.
- NoSQL được sử dụng ở rất nhiều công ty, tập đoàn lớn, ví dụ như Facebook sử dụng Cassandra do Facebook phát triển, Google phát triển và sử dụng BigTable,....

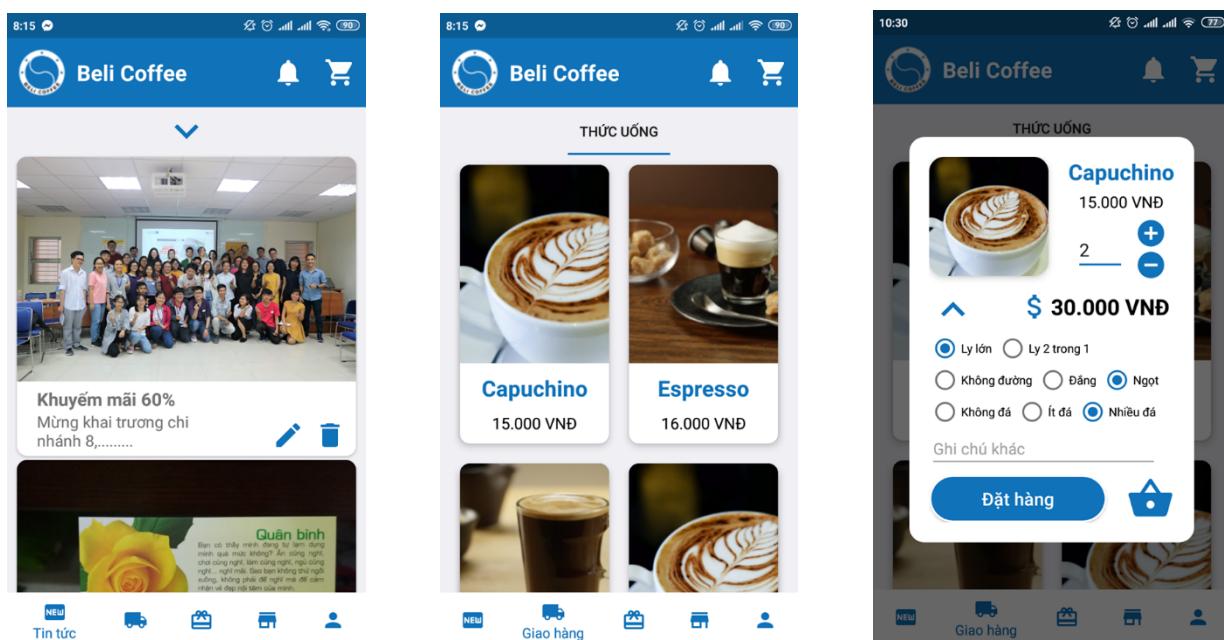
iii. MongoDB

- MongoDB là một hệ quản trị cơ sở dữ liệu mã nguồn mở, là CSDL thuộc NoSQL và được hàng triệu người sử dụng.
- MongoDB là một database hướng tài liệu (document), các dữ liệu được lưu trữ trong document kiểu JSON thay vì dạng bảng như CSDL quan hệ nên truy vấn sẽ rất nhanh.
- Với CSDL quan hệ chúng ta có khái niệm bảng, các cơ sở dữ liệu quan hệ (như MySQL hay SQL Server...) sử dụng các bảng để lưu dữ liệu thì với MongoDB chúng ta sẽ dùng khái niệm là collection thay vì bảng
- So với RDBMS thì trong MongoDB collection ứng với table, còn document sẽ ứng với row, MongoDB sẽ dùng các document thay cho row trong RDBMS.
- Các collection trong MongoDB được cấu trúc rất linh hoạt, cho phép các dữ liệu lưu trữ không cần tuân theo một cấu trúc nhất định.

- Thông tin liên quan được lưu trữ cùng nhau để truy cập truy vấn nhanh thông qua ngôn ngữ truy vấn MongoDB.

The screenshot shows the MongoDB Atlas web interface. At the top, it says "mongoDB. Atlas All Clusters". Below that, "CONTEXT" dropdown is set to "Project 0". The main header "BACHS ORG - 2019-08-26 > PROJECT 0 > CLUSTERS" and "Cluster0" are visible. On the left, a sidebar has sections like "ATLAS", "Clusters" (selected), "Data Lake BETA", "SECURITY", "Database Access", "Network Access", "Advanced", "PROJECT", "Access Management", "Activity Feed", "Alerts", "Settings", and "SERVICES". The main area shows "DATABASES: 1 COLLECTIONS: 1" with a "test.products" collection. It lists "COLLECTION SIZE: 1.13KB TOTAL DOCUMENTS: 4 INDEXES TOTAL SIZE: 36KB". Below this, there are tabs for "Find", "Indexes", and "Aggregation". A search bar contains the filter {"filter": "example"}, and a "Find" button is present. The results section shows "QUERY RESULTS 1-4 OF 4" with one document snippet: "_id: ObjectId(\"5d67a8c8f5bf00004c634b2\"), title: \"Khuyến mãi 60%\", content: \"Những khai trương chi nhánh 8,...\", image: \"https://firebasestorage.googleapis.com/v0/b/belicoffee.appspot.com/o/I...\", __v: 0". A "REFRESH" button is at the top right of the results table.

Hình 17: Giao diện quản lý database của MongoAtlas



Hình 18: Màn hình tin tức và chọn món

III. KẾT LUẬN

1. Hướng phát triển

- Hoàn thiện ứng dụng và đưa lên hệ điều hành Android, bổ sung thêm phiên bản trên hệ điều hành iOS.
- Phát triển ứng dụng chạy trên đa nền tảng (React Native hoặc Flutter).

2. Thuận lợi

- Nguồn tài liệu tham khảo phong phú và đa dạng.
- Đam mê, yêu thích và ham tìm tòi học hỏi.
- Sự hỗ trợ nhiệt tình từ người hướng dẫn cũng như đơn vị thực tập.

3. Khó khăn

- Về kiến thức: Chưa có kiến thức về lập trình di động và mạng máy tính nên gây ra khó khăn trong quá trình tìm hiểu cũng như thực hiện đề tài.
- Về ngoại ngữ: Tài liệu đa số đều là tiếng Anh, và có nhiều từ ngữ chuyên ngành nên gây ra khó khăn trong quá trình đọc hiểu cũng như lập trình.

4. Kinh nghiệm đạt được

- Hiểu được kiến trúc hệ điều hành Android.
- Biết được kiến thức về mạng máy tính và viễn thông.
- Viết được các ứng dụng trên hệ điều hành Android.
- Tăng cường khả năng hòa nhập nhóm, với môi trường làm việc thực tế và kết nối với đồng nghiệp.
- Nâng cao vốn tiếng Anh, kỹ năng thuyết trình và sự tự tin trước đám đông.

TÀI LIỆU THAM KHẢO

[1] Tài liệu về *Android*

<https://developer.android.com/docs>

[2] Tài liệu về *Firebase*

<https://firebase.google.com/docs>

[3] Tài liệu về *WebRTC*

<https://webrtc.org/start/>

[4] Tài liệu về *Node.js*

<https://nodejs.org/en/docs/>

[5] Tài liệu về *MongoDB*

<https://docs.mongodb.com/>